APPLICATION

FOR

UNITED STATES LETTERS PATENT

APPLICANT NAME EDWARD B. BODEN

TITLE SYSTEM AND METHOD FOR IP PACKET FILTERING BASED ON NON-IP PACKET TRAFFIC ATTRIBUTES

DOCKET NO. END920010019US1

INTERNATIONAL BUSINESS MACHINES CORPORATION

CERTIFICATE OF MAILING UNDER 37 CFR 1.10

I hereby certify that, on the date shown below, this correspondence is being deposited with the United States Postal Service in an envelope addressed to the Commissioner of Patents and Trademarks, Washington, D.C., 20231 as "Express Mail Post Office to Addressee" on 07/30/01

Mailing Label No. EL598673254US

Name of person mailing paper: June M. Mitchell

June M. Mitchell 7/30/01

Signature Date

a:formal2.lwp

SYSTEM AND METHOD FOR IP PACKET FILTERING BASED ON NON-IP PACKET TRAFFIC ATTRIBUTES

Background of the Invention

Technical Field of the Invention

This invention pertains to control and management of communication traffic. More specifically, it relates to the use of filtering in the control and management of communication traffic.

Background Art

Referring to Figure 1 in connection with Figure 2, in a network system including two nodes 30, 32 connected by, for example, the Internet 40, the nodes are typically connected through router 35, 37 firewalls 36, 38, respectively. The case is commonly encountered where traffic between nodes 30, 32 is to be restricted based on something other than data in the packet 140. For most business, not all applications are created equal: some data deal with sensitive matters, and usage is restricted to groups or individuals. There is, in

this instance, a need to restrict certain types of traffic by group or application, and the data required to do so is typically not available in packet 140.

Internet protocol (IP) packet filtering, in the context of the TCP/IP protocol suite, is the base technology commonly implemented in firewalls 36, 38 and related systems. It is important for network integrity and security. Virtual private network (VPN) technology based on the Internet protocol security (IPsec) protocols also utilizes packet filtering. VPNs are an important enabler for e-business, both for consumer-to-business (C2B) and business-to-business (B2B) activity.

C2B and B2B applications of VPN technology today must rely on attributes actually contained within the IP packets 140 for their security, or be handled at the application level. This is because IP packet filtering is done based on data actually contained within the datagram 140, either the IP header 142, ULP headers 144, or possibly, the payload portion 154 of the packet.

Since packet filtering does not provide the necessary capabilities, if additional control and management of

aspects of communication traffic are necessary, three techniques are used: (1) application level control, (2) proxy server control, or (3) add packet data control. According to (1), controls are built into the application, such as server application 42. In this case, server application 42 includes configuration capability to control its traffic based on user ID or profile. For example, individual applications 42 may have their own user authorization lists or exit programs allowing control over what users may do. This approach is not centralized and is not standard. Each application may do control differently, and each application must be configured. According to (2), a proxy server 34 controls applications that go into a node 30 from node 32. An application or set of applications or servers is be 'front-ended' by a proxy server 34 which intercepts traffic for its configured applications and performs checking on what users may access. Proxy server 34 creates what looks like node 30, to node 32, but passes through to node 30 only authorized traffic, other traffic being blocked or discarded with, perhaps, an error message. This technique centralizes things somewhat, but has performance problems because traffic comes in from router 35 and up through the stack to an application layer, the proxy 34 has to pretend that it is an application, and then send data back down the stack, and the real server 30 must do the

same. Both techniques (1) and (2) address the problem of data required for control logic in the application not existing in the datagram (aka packet), so the control decision is moved up in the application layer where the required data normally resides. The application layer typically knows the user ID which is requesting data. According to (3), additional required data 152 is added in the packet 140, and this allows filter rules to be written. Thus, application level information 152 may be added to a variety of IP packets 140, so that these decisions can be made about the traffic. However, this makes the packets non-standard, in the sense that these special headers require special function associated with the stack..

The above control and management techniques have various deficiencies, including lack of centralized control when spread over multiple applications and servers, increased overhead when the entire protocol stack must be traversed by IP packet traffic in the case of application control 42 or proxy servers 34, lack of consistency when applying control files, authorization lists, etc. for different applications or servers from different vendors, and lack of security.

It is an object of the invention to provide an improved

system and method for control and management of aspects of communication traffic.

It is a further object of the invention to provide a system and method for control and management of aspects of communication traffic within filtering.

It is a further object of the invention to provide a system and method for centralizing communication management and control within filter rules.

It is a further object of the invention to provide a system and method having reduced overhead for controlling and managing communication traffic, without requiring that IP packet traffic traverse the entire protocol stack to be disallowed.

It is a further object of the invention to provide a system and method having improved consistency, with all the rules for access expressed in the same way.

It is a further object of the invention to provide a system and method for managing and controlling communication traffic having improved security through visibility and coherence by centralizing the access rules and centralizing

associated logging.

Summary of the Invention

A system and method for control and management of communication traffic. Access rules are expressed as filters referencing system kernel data; for outbound processing, source application indicia is determined; for inbound packet processing, a look-ahead function is executed to determine target application indicia; and responsive to the source or target application indicia, filter processing is executed.

In accordance with an aspect of the invention, there is provided a computer program product configured to be operable to control and manage communication traffic, according to the steps of expressing access rules as filters referencing system kernel data; for outbound processing, determining source application indicia; for inbound packet processing, executing a look-ahead function to determine target application indicia; and responsive to said source or target application indicia, executing filter processing.

Other features and advantages of this invention will become apparent from the following detailed description of the presently preferred embodiment of the invention, taken in conjunction with the accompanying drawings.

Brief Description of the Drawings

Figure 1 illustrates various prior art network configurations.

Figure 2 illustrates the basic structure and format of an IP packet.

Figure 3 illustrates the format of a filter statement in accordance with a preferred embodiment of the invention.

Figure 4 illustrates a typical network system adaptable to the present invention.

Figure 5 is a flow diagram illustrating the method steps of an illustrative embodiment of the invention.

Figure 6 is a high level architecture drawing illustrating how the look-ahead function of an illustrative

embodiment of the invention fits within the TCP/IP protocol stack architecture of a system node 30, 32 of Figure 4.

Figure 7 illustrates key elements and the logical relationships and data flow among them for translating FILTER statements to a 6-tuple representation and interpreting them as IP datagrams flow through the OS kernel 120. See Figure 1 of U.S. Patent 6,182,228.

Best Mode for Carrying Out the Invention

IP packet filtering refers to a process that occurs in an operating system kernel implementation of, for example, the TCP/IP protocol suite. Referring to Figure 3, as an extension to normal IP protocol functions, filtering examines each packet 140 by comparing it against a set of filter rules 166. Each filter rule 166 contains a set of 'selectors' 160, each of which specifies a certain field in the packet 140 and some set of values 164, and an operator 162 for performing a match operation. A filter rule 'matches' a packet 140 (and the packet 'matches' the filter rule) when each of that filter's selectors, when checked against the packet, returns 'true'. (A filter rule may have

0 or more selectors, each with associated operator and value set.) If the packet 140 matches a filter rule 166, some action 165 specified by the filter rule is taken. Typical actions include 'deny' (discard the packet), 'permit' (allow the packet to continue) or 'ipsec' (perform some IPsec operations for VPNs on the packet). (Other ancillary actions may be specified by a filter rule, such as logging (aka journaling) the filtering event.)

Acronyms used in the specification include:

IP	Internet Protocol
ID	Identifier
sec	security
VPN	Virtual Private Network
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ICMP	Internet Control Message Protocol

In accordance with a preferred embodiment of the inventions the syntax of a system's filter rule statements 166 are defined to include new parameters (new instances of 160, 162, and 164), and in this manner to extend IP packet filtering to allow filtering on non-IP packet 140 attributes. As illustrated in Figure 3, a filter rule 166

comprises 0 or more of the elements 160, 162 and 164 and a single, non-optional action 165. These extended attributes include, but are not limited to, user ID, user ID attributes, user group, user group attributes, job ID, job name, job attributes, job class and job class attributes. Some details would vary per system, like the preferred name of the parameter or how its values are specified, but the general idea holds everywhere. In general, any data in the operating system kernel, either context data regarding the packet itself (e.g., time-of-day, or on what interface the packet arrived), or data specific to the task (thread, or 'job', or 'user-space process') that generated or will receive the packet, is available for this kernel data extension to IP filtering technology.

This is accomplished by a) extending the syntax used to write filter rules to allow specification of these non-IP packet attributes, and b) extending the filtering function which runs in the system kernel 30 to handle these new attributes. The key to effective implementation is that the filtering code executing in the system kernel 30 would have access to this non-IP packet data via access to non-TCP/IP protocol stack portions of the system kernel 30.

So, for example, a VPN tunnel is typically set up today

between two IP addresses. Current capability also allows specification of port numbers, which would limit traffic to the application bound to those ports, at each end of the VPN tunnel. Now, with these new filtering attributes, the traffic could be controlled further, to ensure that only a particular job, or class of jobs, or particular user ID, or user group, could send and receive data using the VPN tunnel.

This gives administrators the ability to effectively implement system-wide communication controls, based on user-group attributes, job attributes, etc. For example, platinum credit card holders and gold credit card customers could be in different user groups. Hence, different characteristics for these groups is available via IP filtering. And, of course, the filters may be generated for the customer, based on high-level configuration.

Several advantages accrue. For example, the advantages of using packet filtering rather than building controls into the application are that applications 31 don't have to change, controls are centralized in the system 30 in the filter rules, and filtering technology is typically delivered with infrastructure essential to security context such as logging, auditing, controls over filter rule load,

and so forth.

Referring to Figure 3, the phrase 'defined to include new parameters' means; change the syntax of filter statements 166 as currently expressed, to accept each of the new listed parameters in the form of a selector. Each selector allows the specification of selector field 160, an operator 162, and a set 164 of values. The selector field 160 is commonly termed the 'parameter'. Details on selector value sets is not part of the invention because specifics vary across systems.

In existing filter statements, all current selector fields 160 allowed refer to some field of an IP packet 140. These new selector fields, in contrast, refer to data that does not exist in IP packets. (Does not exist in TCP/IP packet headers. Some of these selector fields may, in some cases, appear in the data portion 154 of an IP packet. But since this is very application-specific, and nonstandard, subject to change at any time (etc.), it is not used for filtering, except in special cases.)

A representative (and non-exhustive) list of the new parameters for selector fields 160 is as follows:

userid

user profile

user class

user group

user group authority

user special authority

job name

process name

task name

job group

job class

job priority

other job or process attributes

date & time

(Again, these are representative names, because they vary across systems. A particular representative name may not apply to a given system. But all servers, application platforms and client systems embed these or similar concepts.)

Following are some examples of how selected parameters could be represented in filter statement parameters. (Again, specific syntactical details will vary from platform to platform, but these are not relevant to the invention.)

Filter statements are part of the overall user interface to the system.

1. filter set eth01 destip=p.q.r.s ... usergroup = { staff1, admin2 }
2. filter set eth01 sourceip=a.b.c.d ... jobname = designserver1
3. userprofile = { georgeat, trk, besmith }
4. filter set eth01 destip=a.b.c.d ... date = 5/1-30/2001
5. filter set eth01 inbound ... specialauthority = *iosyscfg

The filtering system and method of the preferred embodiment of the invention exists and executes within node 30, 32 hosts, at least in part, where tests involve getting required information for data available in the kernel.

For outbound packet processing, say from application 31 outbound through node 30 to node 32, a packet 140 is received in the kernel of the operating system of node A 30. This packet is processed by determining the task (or thread) ID, based on the task (or thread) ID determining work control block, from the work control block, determining a process or job identifier, from that identifier determining user space job or process attributes. An alternative path

involves determining the user ID from the work control block, and from the user ID control block determining the attributes for that user.

For inbound packet processing, there is an additional initial step required to determine the target application. Whereas in outbound processing, typically some kernel thread is running on behalf of a user application or process that is sending data out and that thread is readily determined, that is not true on inbound. An inbound datagram has nothing to do with a currently running application. Once that application is determined, then inbound packet processing continues as above described for outbound processing.

To determine the target application 31 for inbound processing, a look-ahead function is executed. According to that look-ahead function, the filter function asks the sockets layer to identify the application 31 to which it would give the packet, but that packet is at this point in the process marked as non-deliverable. This look-ahead has a minor runtime cost (say, in CPU cycles), but much less than executing a proxy server 34.

<u>Outbound packet processing</u>

Referring to Figure 5, in accordance with a preferred embodiment of the invention, steps 60-74 illustrate the method steps executed in support of outbound packet 60 processing. Low-level details on these steps are necessarily system-specific since they involve internal operating system kernel implementation details. However, the steps shown are generally applicable. These steps would be executed as part of kernel IP filtering function, when that filtering function encountered one of the new selector fields (listed above) in a filter statement.

In step 62, for outbound packet, determine the task or kernel thread identifier. This will the task of the caller for the filtering function.

In step 64, locate the kernel control block for that task.

In step 66, locate, via the task control block, the user-space process or job id associated with the task.

In step 68, locate the kernel control block for the user-space process or job id.

In step 70, read, as needed (depending on which

selector field is being processed) the needed information from the job or process control block.

In steps 72 and 74, if required, locate the profile or user id control block associated with the job or process and read the needed information concerning the user from this control block.

For both inbound and outbound filters, a cache of recently referenced non-IP data can be considered so as to lesson the CPU cost of accessing the non-IP data.

Referring further to Figure 5, in connection with Figure 6, which describes how the look-ahead function described hereafter fits within the TCP/IP protocol stack 80, architecture inbound packet 50 processing differs from outbound packet 60 processing primarily in that the task of the caller of filtering is not identified to a user-space 92 process 81, 82, etc., that should receive packet 50. This inbound processing only applies to packets destined for the system doing the filtering, and requires a new IP-level function 100.

In step 52, from the filtering function 90 at the IP-layer 99, invoke a new transport-layer 96 TCP 83, UDP 84, or

RAW 85 'look-ahead' (LA) function 87-89, respectively, by passing the transport-layer 96 header and, if necessary, also the IP protocol 99 header. Which transport-layer 96 look-ahead function 83-85 to call is determined by the IP layer 99 header ip_p field. The selected look-ahead function 83-85 determines which user-level process or job 81-82,..., will receive the packet later, when the packet is sent to the transport layer 96. The task id of the corresponding process 81-82 is returned.

Steps 62-74 described above for outbound processing are now invoked, and function for inbound as well as outbound.

U.S. Patent 6,182,228 describes how to generate filtering code that executes in the operating system (OS) kernel 120 from customer-entered rules.

Referring to Figure 7, which is Figure 1 of U.S. Patent 6,182,228, the key elements and logical relationships and data flow among them, are illustrated for translating FILTER statements 100 to a 6-tuple representation 124, and interpreting them as IP datagrams flow through the OS kernel 120. FILTER (and other rule) statements 100 are processed by rule compiler 102. An output of a first invocation 104 of the rule compiler 102 is two sets of files, s-rule files

106 and i-rule files 108. These files 106, 108 contain the binary form of the rules in image format (in i-rule files 108) or retain some symbolic information (in s-rule files 106). An 'i' or 's' rule file 106, 108 is generated for each physical interface for which there are rules. Later, when the interface is started in response to start TCP interface (STRTCPIFC) command processing 110, a second invocation 114 of rule compiler 102 completes resolution of s-rule files 106. As is represented by step 122, the resolved rules are loaded to OS kernel 120 in the form of 6-tuples. A key part of loading in the kernel is to resolve the various relative and symbolic addresses in 6-tuples to absolute addresses. Thereupon, 6-tuples 124 are ready to be used by filter interpreter 126 as IP datagrams enter and leave the system via device drivers 130 to input/output processor (IOP), not shown. In a specific embodiment, IOPs provide the actual physical interface to a system, and a network cable of some type plugs into an IOP. Each IOP is represented within the OS kernal by a physical interface control block. Filter interpreter 126 communicates with such IOPs through device driver code 130 residing within kernal 120. Transport protocols 128 (such as TCP, UDP) are accessed by filter interpreter 126 is processing 6-tuples 124.

Both image rules (irules) 108 and symbolic rules (srules) 106 are in 6-tuple form, which is the output of rule compiler invocation 104, which is further described in connection with Table 2 of U.S. Patent 6,182,228, above. Both irules 108 and srules 106 go through an address resolution during load 122, of the first two elements of their tuples. However, the srules 106 have an additional resolution that occurs during the compiler call 114 in connection with 122 load, but outside of kernal 120. That additional resolution is required to change certain of their value1 206 or value2 208 elements from symbolic to actual values (IP addresses.)

Advantages over the Prior Art

It is an advantage of the invention that there is provided an improved system and method for control and management of aspects of communication traffic.

It is a further advantage of the invention that there is provided a system and method for control and management of aspects of communication traffic within filtering.

It is a further advantage of the invention that there

is provided a system and method for centralizing communication management and control within filter rules.

It is a further advantage of the invention that there is provided a system and method having reduced overhead for controlling and managing communication traffic, without requiring that IP packet traffic traverse the entire protocol stack to be disallowed.

It is a further advantage of the invention that there is provided a system and method having improved consistency, with all the rules for access expressed in the same way as filters.

It is a further advantage of the invention that there is provided a system and method for managing and controlling communication traffic having improved security through visibility and coherence by centralizing the access rules and centralizing associated logging.

Alternative Embodiments

It will be appreciated that, although specific

embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without departing from the spirit and scope of the invention. In particular, it is within the scope of the invention to provide a computer program product or program element, or a program storage or memory device such as a solid or fluid transmission medium, magnetic or optical wire, tape or disc, or the like, for storing signals readable by a machine, for controlling the operation of a computer according to the method of the invention and/or to structure its components in accordance with the system of the invention.

Further, each step of the method may be executed on any general computer, such as an IBM System 390, AS/400, PC or the like and pursuant to one or more, or a part of one or more, program elements, modules or objects generated from any programming language, such as C++, Java, Pl/1, Fortran or the like. And still further, each said step, or a file or object or the like implementing each said step, may be executed by special purpose hardware or a circuit module designed for that purpose.

Further, the invention applies to any system that supports applications, such as servers or clients, as

distinguished from a system that only performs TCP/IP forwarding & routing functions (that is, systems that do not have any applications, servers or clients).

Accordingly, the scope of protection of this invention is limited only by the following claims and their equivalents.